Continental J. Information Technology 2: 1 - 8, 2008

DETECTION AND COLLECTION OF HOST CONFIGURATION INFORMATION USING MOBILE AGENTS

O. Osunade, Adenike O. Osofisan and O. O. Omoni,
Department of Computer Science, University of Ibadan, Ibadan, Nigeria

ABSTRACT
The job of the network administrator becomes difficult as the number of hosts in a computer network increase. Network management requires the network administrator to have vital configuration information about hosts so as to provide viable solutions. In situations where the hosts span a large area or in several buildings as in tertiary institutions the task is daunting and susceptible to errors. Access to remote and large numbers of hosts can be facilitated using mobile agents. Mobile agents are software that are intelligent, autonomous and mobile. This paper highlights the successful implementation of a mobile agent system for the detection and collection of configuration information of hosts in such a campus-wide network. The mobile agent system was developed and implemented using the Java Development Kit coupled with Visual Basic. The implementation was successful when tested on a local area network of ten computer systems.

KEYWORDS: network administration, information, configuration, Java, mobile agents

INTRODUCTION

There has been tremendous expansion in the area of network deployment in Africa as companies realized the cost benefits and productivity gains created by computer network technology. The technology allowed access to company data, resource sharing, faster communication and ease of document duplication. During the deployment process many hosts were added to the networks and new network technologies and products were introduced.

As the network expanded and grew management of the hosts and the network became increasingly difficult. The management of a network incorporates many tasks such as fault, configuration, accounting, performance and security. The configuration task of network management involves the process of obtaining data from the network and using the data to manage the setup of all network devices. Ensuring that all network devices, such as hosts, and routers, with different rules, are setup and configured properly is a daunting task. Network engineers strive to meet this task by developing a set of operating procedures and policies that meet the needs of their users. To meet users' needs presents an enormous challenge in the face of unprecedented traffic growth, many installed devices, cut-throat competition, vendor hardware and software upgrades, outside attacks by hackers, viruses and an insufficient tool set for configuring large numbers of devices.

The configuration of devices on most networks have been done manually by network engineers (Wolf, 2001) leading to a broad range of operational problems and issues, such as unknown network configurations, security loopholes, lost passwords, absence of audit trails, postponed or irreversible software upgrades, and so on. Also to obtain configuration information from the network requires manual effort as the network administrator may have to physically or remotely login to reach each device on the network and then record the required information about the device e.g. serial number, address. The obtained result is then stored in a flat ASCII file, spreadsheet or database as the case may be.

The pitfalls of the manual approach can be avoided when an automated tool is used for obtaining configuration information of devices connected to the network. This paper therefore focuses on the use of mobile agents for centrally maintaining configuration information about devices such as hosts and routers, on a network. Using mobile agent technology, detailed configuration information about device drivers, disk configuration and application data on a device can be collected and transmitted to a central management repository for administrative and technical uses.

LITERATURE REVIEW

Network management is a service that employs a variety of tools, applications, and devices to assist human network managers in monitoring and maintaining networks (Leinward and Conroy, 1998). Network managers stay in control using these tools to track and monitor all activity that has the potential to affect the network's performance. Network management helps to determine faults, performance, and configuration changes in a data network.

Most network management architectures use the same basic structure and set of relationships. End stations such as computer systems and other network devices, run software that enables them to send alerts when they recognize problems. Upon receiving these alerts, management entities are programmed to react by executing one, several, or a group of actions, including operator notification, event logging, system shutdown, and automatic attempts at system repair. Management proxies are entities that provide management information on behalf of other entities.

Management entities also can poll end stations to check the values of certain variables. Polling can be automatic or user-initiated, but agents in the end stations respond to all polls. Agents are software modules that first compile information about the managed devices in which they reside, then store this information in a management database, and finally provide it to management entities within network management systems (NMSs) via a network management protocol (Raibulet and Demartini, 1999). Well-known network management protocols include the Simple Network Management Protocol (SNMP) and Common Management Information Protocol (CMIP). Fig. 1 depicts a typical network management architecture.

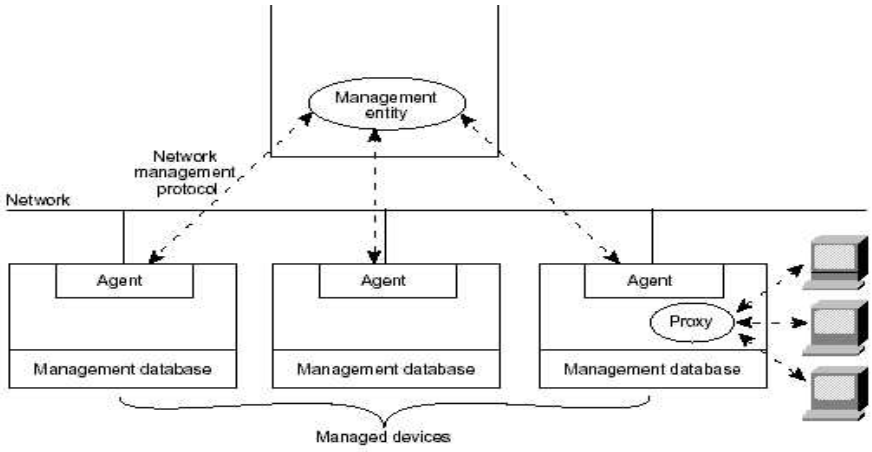

Fig.1: A typical Network Management Architecture maintains many Relationships (Source: Plakosh, 2001)

Configuration management is the process of gathering information from the network and using that data to manage and optimize network devices. Common configuration management tasks include the assignment of network addresses to network devices and maintaining an up-to-date inventory of equipment installed on the network. Configuration management is a prerequisite function to performance management. The goal of configuration management is to monitor network and system configuration information so that the effects on network operation of various versions of hardware and software elements can be tracked and managed.

Configuration management subsystems store this information in a database for easy access. When a problem occurs, this database can be searched for clues that may help solve the problem. To get access to the database would require the network manager to adopt either the client-server method or to manually access the database on each system. However if mobile agents are used the management process is simplified and productivity enhanced. Chess *et al* (1997) and Lange and Oshima (1999) gave reasons why mobile agents are suitable for use in network management.

The giant leap of computer communication technology from a few communicating systems to millions of systems geographically dispersed have forced the re-evaluation of distributed systems by extending the applications available. The client-server communication method which had hitherto being used proved inadequate. This necessitated the use of mobile code systems, of which the mobile agent paradigm has been the one with most promise. A complicated task can be decomposed and performed through the cooperation of several distributed agents on separate interconnecting systems each of which may have access to different tools and data, so that it is more efficient.

The mobile agent paradigm is a product of research from two fields of computing: artificial intelligence (AI) and distributed computing. AI uses intelligent computing entities to take over work from human beings and simplify human operations. In AI, this is done by a computer program, called a software agent that helps a user perform some task or a set of tasks (Lingnau *et al*, 1995). An agent is expected to have AI properties to perform its tasks such as act on behalf of, interact with a user, take initiative and reason about activities, communicate with and learn from the environment, and to react to environmental changes (Freeman, 2001).

A software agent is a program that is autonomous enough to act independently, even when the user or application that launched it is not available to provide guidance and handle errors (Kotz *et al*, 1996). All software agents are programs, but not all programs are agents (Feldman and Yu, 1999). A software agent with mobility is called a mobile agent.

According to Gilbert *et al* (1995), software agents can be classified in terms of the three dimensions of intelligence, agency and mobility as shown in Table 1. These dimensions offer different levels of operation to users, with higher levels being in the direction of the arrow.

Table 1. The three dimensions of software agents

Intelligence	Agency	Mobility
Preferences	Asynchrocity	Message passing
Reasoning	Representation of user	Remote procedure call
Planning	Data interactivity	Remote execution
Learning	Application Interactivity	Weak migration
	Service Interactivity	Strong migration

Adapted from Rothermel & Schwehm (1998).

Intelligence in software agents is based on artificial intelligence research which dates back to the fifties, where the term agent was coined by Selfridge for a ‘soft robot’ living and acting within a computer (Rothermel & Schwehm, 1998).The goal of intelligence for software agents was to apply techniques of symbolic artificial intelligence in order to perform a given task or to recover when it was stuck. Intelligent agents can be classified according to their capabilities to express preferences, beliefs and emotions and according to their ability to perform a task by reasoning, planning and learning techniques.

Agency is the degree of autonomy and authority vested in a software agent. This can be measured quantitatively by the nature of interaction between the agent and other entities of the system (Agentbuilder, 2001). The degree of agency is enhanced if an agent represents a user in some way.

The third feature of software agents is mobility. The goal of software agents with mobility is remote action and mobility of data and computation. Mobile agents can be classified according to the scope of their mobility that is desktop or static agents, Intranet agents, Internet agents or network agents.

Network management architectures require configuration information to be stored in databases which can be accessed by mobile agents launched from the network managers system. Thus our paper presents the design and implementation of mobile agents accessing configuration management information.

OUR DESIGN

The network configuration detection and collection system developed incorporates four major components.
The Home Agent
The Probe Agent
The Roaming Agent
The Reporter Module

The Home Agent: This is a back-end static agent that is responsible for creating an environment for the other agents to operate in. This agent resides on every host in the network the mobile agent would migrate to. The home agent has an initial graphical user interface and is responsible for hosting the other agents and instantiating the arriving agents. The home agent is initiated by entering the address (127.0.0.1) into the Internet Protocol (IP) field of the agent launch form.

The Probe Agent: This agent is the major means through which the system gathers and stores the configuration information. It resides in the roaming agent so as to utilize its' mobility. This agent has no graphical user interface. However, it gathers and logs configuration data to be reported in the roaming agent. The probe agent was developed in Visual Basic.

The Roaming Agent: This agent provides mobility within the network. The task it performs is to migrate from one host to the next depending on the host address specified on the server end. It also serves as the vehicle through which the probe agent achieves migration. After a roaming session, the roaming agent returns to the initial host on which it was created to allow the reporter module to make use of the information gathered for reporting purposes.

The Reporter Module: This agent is also a static agent that resides on one of the host on the network. This host serves as the home for the roaming agent i.e. the agent starts out from this host and also reports back after visiting available hosts. This agent provides the system with a graphical user interface for viewing data collected and using the data for reporting purposes. This agent was also developed in Visual Basic.

METHODOLOGY

The design above was implemented for use on a campus-wide local area network. It was simulated on a local area network of ten computer systems in the Department of Computer Science, University of Ibadan, Nigeria. All the computers used had the Java Virtual Machine (JVM) installed or the JDK (Java Development Kit).

The programming languages used for this project work were Java and Microsoft Visual Basic. Java was used to implement mobility in the system i.e. the home agent and the roaming agent were developed using Java. The choice of Java for this part of the system was because of the ability of Java to implement socket operations and object serialization (Jackson and McClellan, 1997). Another major consideration was the platform independence nature of Java. Since the application is designed to run on computer networks, the demand for security must also be met. Java's security architecture therefore makes Java a more suitable development platform for mobile agent development

The implementation of the probe agent and the reporter module was in Visual Basic. This language was chosen because of the ability of the programming language to communicate with the computer system to extract system information which is a feature lacking in Java due to its platform independence nature. Another reason for the selection of Visual Basic was because of its ability to report data using custom tools providing a versatile reporting system that can be used by system administrators.

IMPLEMENTATION RESULTS AND DISCUSSION

The welcome screen in Fig. 2 presents the opportunity for the actual launching of the roaming agent by clicking the *launch agent button* on the graphical interface. The agent launcher screen shown in Fig. 3, then allows the user to specify the Internet Protocol (IP) address of the host to send the roaming agent to. The specified host must have the home agent running and ready before this connection can be made.

The default IP supplied in this dialog is to initiate the server to act as a host in special cases when the server is required to serve in such a capacity. To initiate the system to deploy the roaming agent the default IP is changed to that of the host to visit. The Launch Agent button is then clicked and the Roaming agent attempts to migrate to the specified host. At its arrival at the destination the agent proceeds to execute on the host to

Fig. 2. Welcome and menu screen for the Roaming Agent

Fig 3. Agent Launcher screen

which it migrates and after finishing its defined functions it reconnects with the server from which it originated and migrates back to this server computer to report the result gathered from the just concluded tour. The progress of the agent is also shown on a command prompt console as this proceeds.

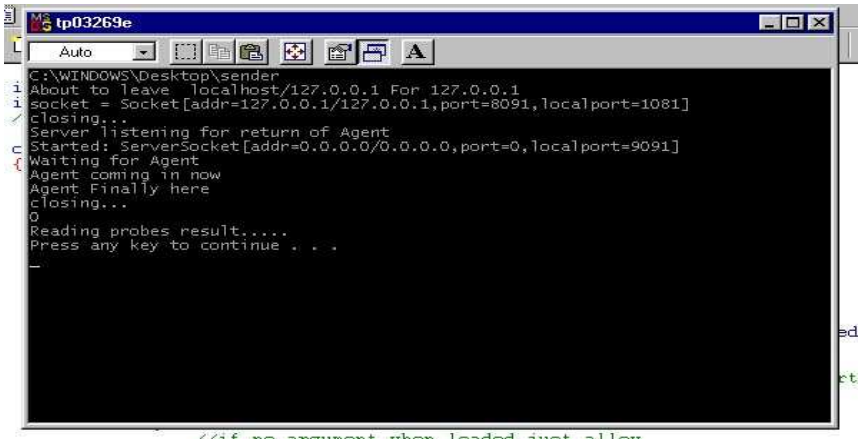

Fig 4. The Roaming Agent Progress Screen

When the home agent is initialized on a host it waits for the roaming agent from the server to migrate into the domain. On arrival the home agent initializes the incoming agent object and ensures that the roaming agent completes its operation. The probe agent runs as a subsystem of the roaming agent and does not have any interface. On arrival at the host, the probe agent is decoupled and executed to detect configuration information after which it terminates.

After the roaming agent has gathered the required information via the probe agent, the home agent provides the ability for the roaming agent and the data detected and collected to return back to the server from which it was launched. The activities of the home agent and probe agent are indicated through the command prompt screen as shown in Fig. 5.

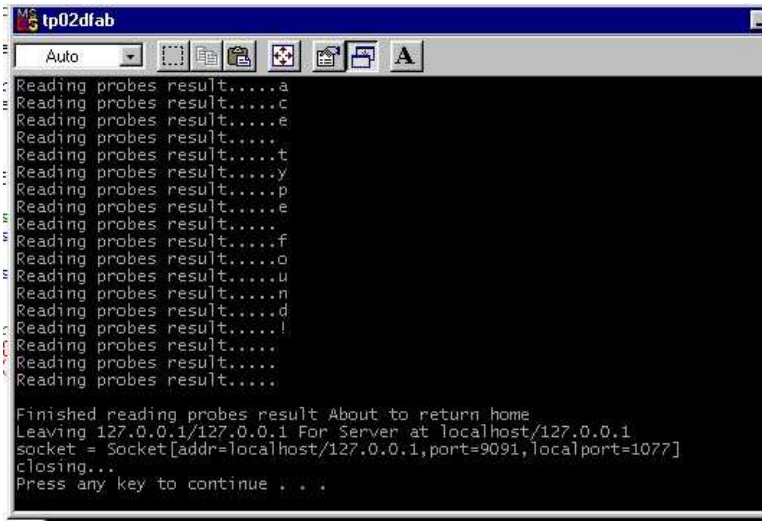

Fig 5. Home Agent Screen

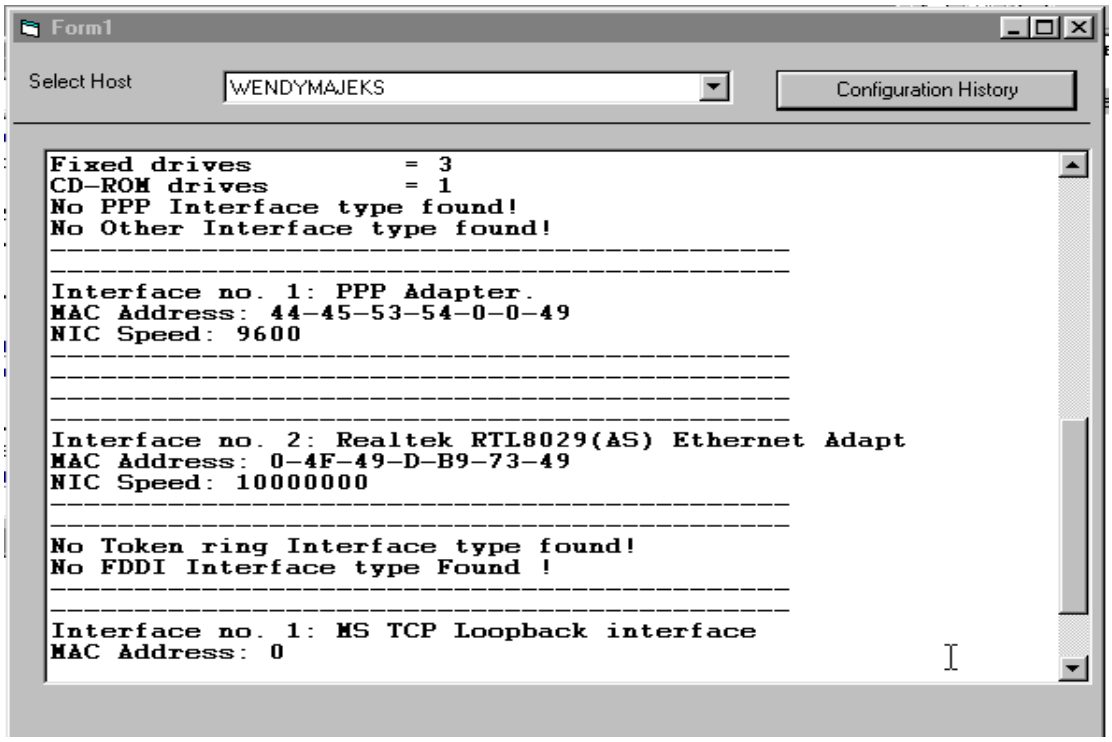

Fig 6. Reporter Module Screen

The reporter module scans the log file for host names and allows the user to select a host for which a report is generated. Upon selection the lower pane of the report screen, Fig. 5, shows the information gathered from the log file for the host e.g. WENDYMAJEKS, selected by the user. The user or Network Administrator can then use the detected configuration (system or network) for administration and planning purposes.

CONCLUSION

Mobile agent technology is a promising communication paradigm that allows tasks such as network management to be carried out autonomously, easily and with minimal physical or technical resources. The detection and collection of configuration information about hosts on a network is useful to the network administrator who usually has many hosts to manage. The implementation of the mobile agent system using Java and Visual Basic provided the ability to obtain the required system information.

REFERENCES

Agentbuilder (2001): *Why, When, and Where to Use Software Agents.* Retrieved October 24, 2007 from http://agentbuilder.com/Documentation/whyAgents.html

Chess, D.; Harrison, C. and Kershenbaum, A. 1997. Mobile agents: are they a good idea? *Mobile Object Systems: Towards the Programmable Internet, (MOS'96), Australia*. J. Vitek and C. Tschudin, eds. Lecture Notes in Computer Science, Springer-Verlag, 1222:25-45.

Feldman, S. and Yu, E. 1999. Intelligent agents: a primer. *Searcher*, 7.9. Retrieved July 20, 2005 from http://www.infotoday.com/searcher/oct99/feldman+yu.htm

Freeman, Y.F. 2001. The spider model of agents. M.Sc.Thesis. Dept. of Computing and Information Science, Queen's University, Ontario, Canada. vi+84.

Gilbert, D.; Aparicio, M.; Atkinson, B.; Brady, S.; Ciccarino, J.; O'Connor, P.; Osisek, D.; Pritko, S.; Spagna, R. and Wilson, L. 1995. *IBM intelligent agent strategy*. IBM Corporation.

Jackson, J.R. and McClellan, A.L. (1997): *Java by example*. 2nd Edition, SunSoft Press, U.S.A.

Kotz, D., Gray, R., Rus, D., Nog, S. and Cybenko, G. 1996. *Mobile agents for mobile computing.* Technical Report PCS-TR96-285. Retrieved July 20, 2005 from http://www.cs.dartmouth.edu/~rus/papers/agents/mobile.ps.Z

Lange, D.B. and Oshima, M. 1999. Seven good reasons for mobile agents. *Communications of the ACM*, 42.3:88-89.

Leinwand, A. and Conroy, K. F. (1998): *Network Management: A practical perspective.* 2nd Edition, Addison Wesley Longman Inc, Harlow, England.

Lingnau, A.; Drobnik, O. and Domel, P. 1995. An HTTP-based infrastructure for mobile agents. *Proceedings of the 4th International WWW Conference, USA.* O'Reilly and Associates, 461-471.

Plakosh, D. (2001): *Network Management – An Overview. Retrieved October 24, 2007 from http*://www.sei.cmu.edu/str/descriptions/network_body.html

Raibulet, C. and Demartini, C. (1999): *Mobile Agent Technology for the Management of Distributed System-a case study.* Retrieved October 24, 2007 from http://www.terena.nl/tnc2000/proceedings/1A/1a2.html

Rothermel, K. and Schwehm, M. 1998. Mobile agents. *Encyclopedia for Computer Science and Technology*. A. Kent and J.G. Williams, eds. M. Dekker Inc.

Wolf, J. (2001): *Configuration management: Automating deployment of services throughout network.* Retrieved October 24, 2007 from http://www.unisysworld.com/monthly/2001/06/configmgmt.shtml

Received for Publication: 21/12 /2007
Accepted for Publication: 02/02 /2007

Corresponding Author:
O. Osunade,
Department of Computer Science, University of Ibadan, Ibadan, Nigeria
Email: o.osunade@mail.ui.edu.ng